Understanding SSH

SSH is a cryptographically protected remote login protocol that replaces insecure telnet and rlogin protocols. It provides strong protection against password sniffing and third party session monitoring, better protecting your authentication credentials and privacy. In addition, SSH offers additional authentication methods that are considered more secure than passwords, such as public key authentication and extensive protection against spoofing.

Authentication in SSH

SSH servers offer the client a selection of authentication methods. The server advertises what it supports, and the client attempts to authenticate over each method that it can support. Generally, the client will choose methods that are the least intrusive to the user, if they are available. In most cases, the client provides the option to choose which methods can be used.

Verifying the Host Key

If you are using an SSH client to connect to a server for the first time, you will probably see a message looking something like this:

```
The server's host key was not found in the cache. You have no guarantee that the
server is the computer you think it is.
The server's rsa2 key fingerprint is:
ssh-rsa 1024 94:3c:9e:2b:23:df:bd:53:b4:ad:f1:5f:4e:2f:9d:ba
```

This is a feature of the SSH protocol. It is designed to protect you against a network attack known as spoofing: secretly redirecting your connection to a different computer, so that you send your password to the wrong machine. Using this technique, an attacker would be able to learn the password that guards your login account, and could then log in as if they were you and use the account for their own purposes.

To prevent this attack, each server has a unique identifying code, called a host key. These keys prevent a server from forging another server’s key. If you connect to a server and you receive an unexpected host key, an SSH client can warn you that the server may have been switched and that a spoofing attack might be underway.

An SSH client records the host key for each server you connect to. Every time you connect to a server, it compares the server's host key to the host key you received the last time you connected. If the keys differ, you will receive a warning and a chance to abandon your connection before you enter any private information such as a password.

However, when you connect to a server for the first time, an SSH client has no way of telling whether the host key is the right one or not. So it gives the warning shown above, and asks you whether you want to trust this host key or not.

Whether or not to trust the host key is your choice. If you are connecting within a company network, you might feel that all the network users are on the same side and spoofing attacks are unlikely, so you might choose to trust the key without checking it. If you are connecting across a hostile network (such as the Internet), you should check with your system administrator, perhaps by telephone or in person. (Some modern servers have more than one host key. If the system administrator sends you more than one fingerprint, you should make sure the one that the client shows you is on the list, but it doesn't matter which one it is.)

Encryption in SSH

SSH clients and servers can use a number of encryption methods. In the older SSH-1 protocol, 3DES and DES are typically used. SSH-2 adds support for additional encryption methods including AES and Blowfish. By default, AES is used if supported by the server. While AES is considered to be highly secure, AES encryption requires substantial processor overhead. Blowfish is also considered secure, but with less computational overhead, it's also theoretically easier to perform a brute-force attack. Depending on your security and performance requirements, you may wish to configure your client to prefer the Blowfish algorithm. 3DES and DES are used with SSH-1 servers. DES is widely regarded as insecure, as the resources to perform an exhaustive brute-force attack have been well within the realm of commercial feasibility for some time.

SSH Protocols

Two major versions of the SSH protocol are in widespread use. The SSH-1 protocol is an older version that's still widely supported despite its age and some technical issues. The SSH-2 protocol has become the de-facto installation standard, though some systems only support SSH-1. In addition, many sites that use SSH-2 disable the SSH-1 protocol for security reasons.

A client's typical default setting is to prefer SSH-2 and negotiate down to SSH-1 if SSH-2 is not available. If the majority of systems you connect to are using SSH-2, you may wish to change this setting.

Compression

SSH supports data stream compression between the client and the server. On slow links, this may increase throughput, while in faster connections the added CPU overhead may actually result in slower transfers, particularly depending on the data type you're transferring. Large text files may still benefit significantly, while binaries may transfer more slowly. You may want to experiment to find what works best in your situation. Compression may also improve security slightly, in part by rendering known cyphertext attacks more difficult and by providing less data for cryptanalysis.

Adapted from: http://winscp.net/eng/docs/ssh